04 | 升级：Vue 2项目如何升级到Vue 3？

2021-10-25 大圣

《玩转Vue 3全家桶》 课程介绍 >

讲述：大圣

时长 12:31 大小 11.47M

你好，我是大圣，欢迎进入课程导读篇的第四讲。

在上一讲，我带你了解了 Vue 3 的新特性。对于 Vue 3 相比于 Vue 2 有哪些优势这个问题，相信你已经了解得很清楚了。那么在这一讲，我来教你如何把 Vue 2 的项目升级到 Vue 3。

把 Vue 2 的项目升级到 Vue 3，也是小圣一直关心的问题，今天早晨小圣还问我，既然 Vue 3 如此优秀，是不是应该赶紧把项目都升级到 Vue 3？

首先不要着急，并不是所有项目都适合升级。就像苹果出了新款手机，哪怕新特性被人们说得天花乱坠，但是，是不是把老手机换掉，也需要斟酌，毕竟升级总是需要成本的。

应不应该从 Vue 2 升级到 Vue 3

应不应该升级？这个问题不能一概而论。

首先，如果你要开启一个新项目，那直接使用 Vue 3 是最佳选择。后面课程里，我也会带你使用 Vue 3 的新特性和新语法开发一个项目。

以前我独立使用 Vue 2 开发应用的时候，不管我怎么去组织代码，我总是无法避免在 data、template、methods 中上下反复横跳，这种弊端在项目规模上来之后会更加明显。而且由于 vue-cli 是基于 Webpack 开发的，当项目规模上来后，每执行一下，调试环境就要 1 分钟时间，这也是大部分复杂项目的痛点之一。

而 Vue 3 的 Composition API 带来的代码组织方式更利于封装代码，维护起来也不会上下横跳。Vite 则带来了更丝滑的调试体验，一步步跟着专栏完成你的第一个 Vue 3 项目，你会感受到 Vue 3 的魅力。

Vue 3 的正式版已经发布有一年了，无论是辅助工具，还是周边库都已经非常完善了，足以胜任大型的项目开发。并且，现在也有越来越多的公司正在尝试和体验 Vue 3。所以新项目可以直接拥抱 Vue 3 的生态，这也是现在很多团队在做的尝试。

而且对于 Vue 2，官方还会再维护两年，但两年后的问题和需求，官方就不承诺修复和提供解答了，现在继续用 Vue 2 其实是有这个隐患的。

Vue 3 也不是没有问题，由于新的响应式系统用了 Proxy，会存在兼容性问题。也就是说，如果你的应用被要求兼容 IE11，就应该选择 Vue 2。而且，Vue 团队也已经放弃 Vue 3 对 IE11 浏览器的支持。

其实，官方原来是有计划在 Vue 3 中支持 IE11，但后来由于复杂度和优先级的问题，这个计划就搁置了下来。

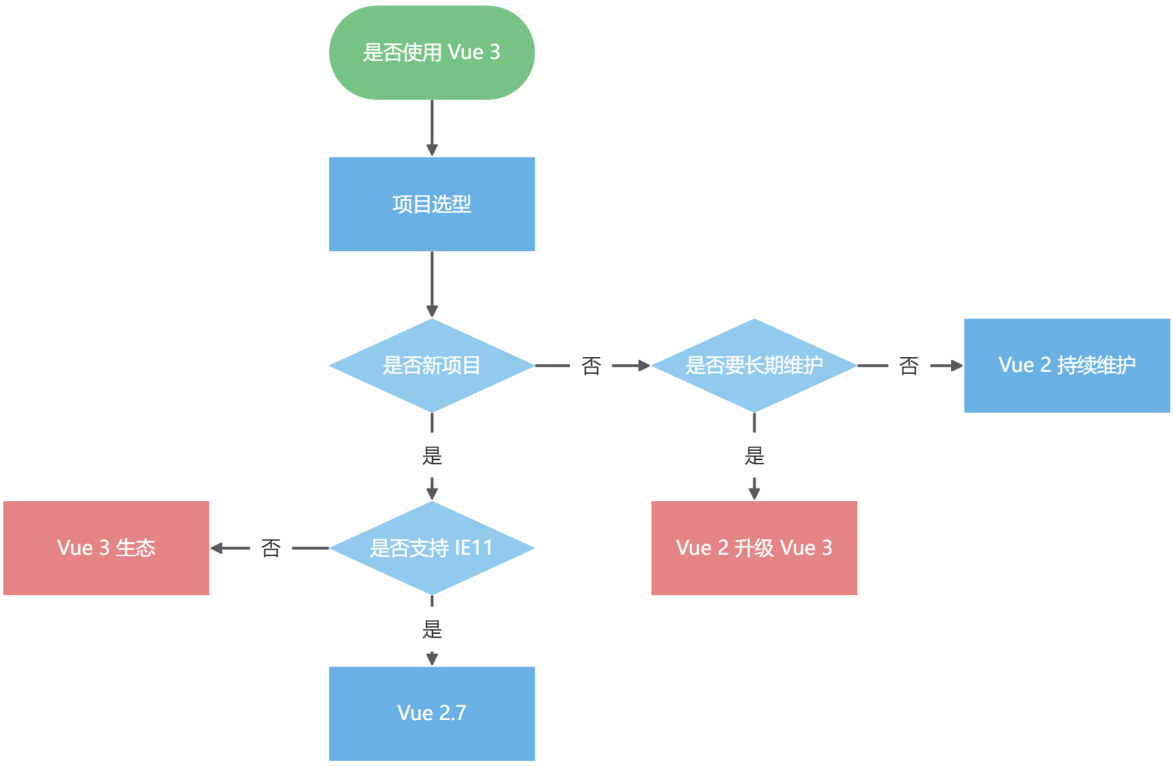
不过，站在 2021 看待现在前端的世界，你能发现浏览器和 JavaScript 本身已经有了巨大的发展。大部分的前端项目都在直接使用现代的语言特性，而且微软本身也在抛弃 IE，转而推广 Edge。所以 Vue 官方在重新思考后，决定让 Vue 3 全面拥抱未来，把原来准备投入到 Vue 3 上支持 IE11 的精力转投给 Vue 2.7。

那么 Vue 2.7 会带来什么内容呢?

Vue 2.7 会移植 Vue 3 的一些新特性，让你在 Vue 2 的生态中，也能享受 Vue 3 的部分新特性。在 Vue 3 发布之前，Vue 2 项目中就可以基于 @vue/composition-api 插件，使用 Composition API 语法，Vue 2 会直接内置这个插件，在 Vue 2 中默认也可以用 Compositon 来组合代码。

后面，我会带你学的 <script setup> 语法，也会在 Vue 2 中得到支持。当然，如果我们想用更精简的方式去组织代码，也是没问题的，因为 Vite 中也正式支持了 Vue 2。

综上所述，我们用下图来回答这一讲开头的问题，要不要使用 Vue 3，还是要“因地制宜”，在不同的场景下，我们选择合适的方式即可。



极客时间

Vue 3 不兼容的那些写法

通过前面的分析，在选择 Vue 2 还是 Vue 3 这个问题上，相信你现在已经有了自己的取舍。如果最后你依然决定要升级 Vue 3，那我就先带你了解一下 Vue 3 不支持的那些写法、之后为你讲解它的生态现状，最后，我们再进入到实操升级的环节。

了解一下 Vue 3 不兼容的那些具体语法，除了可以帮你在升级项目后，避免写的代码无法使用，还会让你更好地适应 Vue 3。详细的兼容性变更，官方有一个迁移指南，我在这里就不一一给出了。同样，也为了避免八股文的形式，我在这里介绍几个重要的变更，后面项目中用到一些写法的时候，我再详细地告诉你。即使现在说太多细节，可能你也记不住。

这一部分内容，主要是针对有 Vue 2 开发经验的，希望更快地适应 Vue 3 的同学。在全面实战 Vue 3 之前，你不必完整阅读官方的指南，因为 Vue 3 的大部分 API 都是对 Vue 2 兼容的。

首先，我们来看一下 Vue 2 和 Vue 3 在项目在启动上的不同之处。在 Vue 2 中，我们使用 new Vue() 来新建应用，有一些全局的配置我们会直接挂在 Vue 上，比如我们通过 Vue.use 来使用插件，通过 Vue.component 来注册全局组件，如下面代码所示：

复制代码

```
Vue.component('el-counter', {
  data(){
    return {count: 1}
  },
  template: '<button @click="count++">Clicked {{ count }} times.</button>'
})

let VueRouter = require('vue-router')
Vue.use(VueRouter)
```

在上面的代码里，我们注册了一个 el-counter 组件，这个组件是全局可用的，它直接渲染一个按钮，并且在点击按钮的时候，按钮内的数字会累加。

然后我们需要注册路由插件，这也是 Vue 2 中我们使用 vue-router 的方式。这种形式虽然很直接，但是由于全局的 Vue 只有一个，所以当我们在一个页面的多个应用中独立使用 Vue 就会非常困难。

看下面这段代码，我们在 Vue 上先注册了一个组件 el-counter，然后创建了两个 Vue 的实例。这两个实例都自动都拥有了 el-couter 这个组件，但这样做很容易造成混淆。

复制代码

```
Vue.component('el-counter',...)
```

```
2
3 new Vue({el:'#app1'})
4 new Vue({el:'#app2'})
```

为了解决这个问题，Vue 3 引入一个新的 API ，createApp，来解决这个问题，也就是新增了 App 的概念。全局的组件、插件都独立地注册在这个 App 内部，很好的解决了上面提到的两个实例容易造成混淆的问题。下面的代码是使用 createApp 的简单示例：

复制代码

```
1 const { createApp } = Vue
2 const app = createApp({})
3 app.component(...)
4 app.use(...)
5 app.mount('#app1')
6
7 const app2 = createApp({})
8 app2.mount('#app2')
```

createApp 还移除了很多我们常见的写法，比如在 createApp 中，就不再支持 filter、$on、$off、$set、$delete 等 API。不过这都不用担心，后面我会告诉你怎么去实现类似这些 API 的功能。

在 Vue 3 中，v-model 的用法也有更改。在后面讲到组件化，也就是我们需要深度使用 v-model 的时候，我会再细讲。 其实 Vue 3 还有很多小细节的更新，比如 slot 和 slot-scope 两者实现了合并，而 directive 注册指令的 API 等也有变化。你现在记不住这些也不要紧，我们会在后面的实战项目里逐渐掌握这些内容。

Vue 3 生态现状介绍

在 Vue 生态中，现在所有官方库的工具都全面支持 Vue 3 了，但仍然有一些生态库还处于候选或者刚发布的状态。**所以，升级 Vue 3 的过程中，除了 Vue 3 本身的语法变化，生态也要注意选择。**有一些周边的生态库可能还存在不稳定的情况，开发项目的时候我们时刻关注项目的 GitHub 即可。

Vue-cli4 已经提供内置选项，你当然可以选择它支持的 Vue 2。如果你对 Vite 不放心的话，Vue-cli4 也全面支持 Vue 3，这还是很贴心的。

vue-router 是复杂项目必不可少的路由库，它也包含一些写法上的变化，比如从 new Router 变成 createRouter；使用方式上，也全面拥抱 Composition API 风格，提供了 useRouter 和 useRoute 等方法。

Vuex 4.0 也支持 Vue 3，不过变化不大。有趣的是 Vue 官方成员还发布了一个 Pinia，Pania 的 API 非常接近 Vuex5 的设计，并且对 Composition API 特别友好，更优雅一些。在课程后续的项目里，我们会使用更成熟的 Vuex4。

其他生态诸如 Nuxt、组件库 Ant-design-vue、Element 等等，都有 Vue 3 的版本发布。我开发维护的 Element3 是一个教育项目，我们在组件化章节会详细介绍。除此之外，我们项目中也会使用 Elemen3 来作为组件库。并且在进阶开发篇，我们会自己设计一个类似风格的组件库。

使用自动化升级工具进行 Vue 的升级

小项目不用多说，从 Vue 2 升级到 Vue 3 之后，对于语法的改变之处，我们挨个替换写法就可以。**但对于复杂项目，我们需要借助几个自动化工具来帮我们过渡。**

首先是在 Vue 3 的项目里，有一个 @vue/compat 的库，这是一个 Vue 3 的构建版本，提供了兼容 Vue 2 的行为。这个版本默认运行在 Vue 2 下，它的大部分 API 和 Vue 2 保持了一致。当使用那些在 Vue 3 中发生变化或者废弃的特性时，这个版本会提出警告，从而避免兼容性问题的发生，帮助你很好地迁移项目。并且通过升级的提示信息，@vue/compat 还可以很好地帮助你学习版本之间的差异。

在下面的代码中，首先我们把项目依赖的 Vue 版本换成 Vue 3，并且引入了 @vue/compat 。

复制代码

```
"dependencies": {
-  "vue": "^2.6.12",
+  "vue": "^3.2.19",
+  "@vue/compat": "^3.2.19"
    ...
},
"devDependencies": {
-  "vue-template-compiler": "^2.6.12"
+  "@vue/compiler-sfc": "^3.2.19"
}
```

然后给 vue 设置别名 @vue/compat，也就是以 compat 作为入口，代码如下：

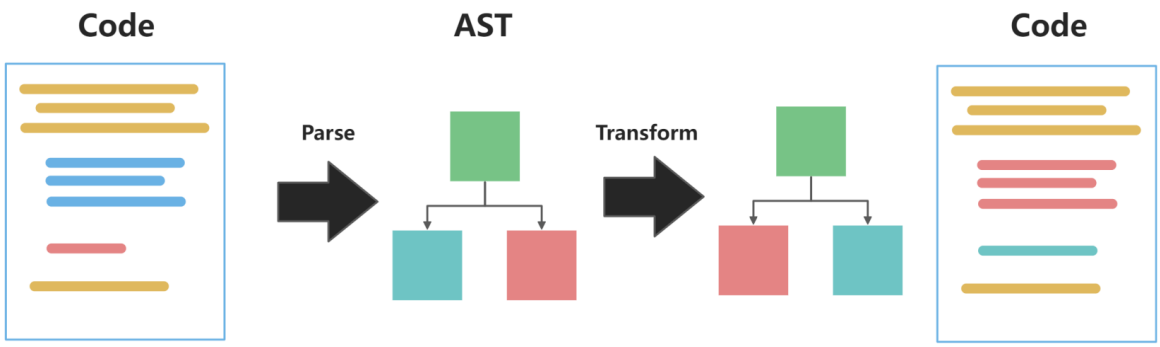
复制代码

```
// vue.config.js
module.exports = {
  chainWebpack: config => {
    config.resolve.alias.set('vue', '@vue/compat')
    ......
  }
}
```

这时你就会在控制台看到很多警告，以及很多优化的建议。我们参照建议，挨个去做优化就可以了。

在 @vue/compat 提供了很多建议后，我们自己还是要慢慢做修改。但从另一个角度看，“偷懒”是优秀程序员的标志，社区就有能够做自动化替换的工具，比较好用的就是“阿里妈妈”出品的 gogocode，官方文档也写得很详细，就不在这里赘述了。

自动化替换工具的原理很简单，和 Vue 的 Compiler 优化的原理是一样的，也就是利用编译原理做代码替换。如下图所示，我们利用 babel 分析左边 Vue 2 的源码，解析成 AST，然后根据 Vue 3 的写法对 AST 进行转换，最后生成新的 Vue 3 代码。



对于替换过程的中间编译成的 AST，你可以理解为用 JavaScript 的对象去描述这段代码，这和虚拟 DOM 的理念有一些相似，我们基于这个对象去做优化，最终映射生成新的 Vue 3 代码。

关于 AST 的细节，在课程后面的 Vue 3 生态源码篇中，我会带你手写一个迷你版的 Vue 3 Compiler，那时你会对 AST 和它背后的编译原理有一个更深的认识。

总结

我们来总结一下你今天都学到了什么吧。**首先，我带你明确了什么时候该升级 Vue 3，什么时候该继续使用 Vue 2 的兼容版本。**现在，Vue 3 的官方生态在整体都比较稳定，新的项目完全可以直接选择 Vue 3。并且，对于那些需要长期维护的项目，其实也很有必要进行升级。不过，Vue 2 很快会停止更新，如果你的项目需要兼容 IE11，那就需要继续使用 Vue 2.7。这样，在保持好项目的兼容性的前提下，还可以体验到 Composition API 带来的便利。

然后，在升级 Vue 的过程中，我们可以利用官方和社区的工具，帮助我们高效地升级。我们可以使用 compat 来给出提醒，项目中设置 @vue/compat 作为 vue 的别名，这样内部就会把所有和 Vue 2 的语法相关的升级信息提示出来，逐个替换即可，或者直接使用 gogocode 进行自动化批量替换。

最后，我想说的是，**全面拥抱 Vue 3 也算是一次离开舒适圈的挑战，这带来的不只是新框架的体验，同时也可能是更好的潜力与更好的待遇。**课程导读篇也到此结束了，在课程的下一篇，我们会开始实战一个 Vue 3 的项目，与此同时，我也会手把手带你通过实战学会 Vue 3。

思考题

你可以分享一下，在你的项目里，有没有需要升级的地方？有的话，又在什么地方呢？

欢迎你在留言区跟我交流讨论，也推荐你把这一讲推荐给你自己的朋友、同事。我们下一讲见！

分享给需要的人，Ta订阅后你可得 20 元现金奖励

生成海报并分享

赞 16 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 03 | 新特性：初探Vue 3新特性

精选留言 (14)

写留言

cwang 酥

2021-10-25

谢谢大圣老师的讲解，我还有一个小的疑惑，我们通常创建Vue应用，一个实例（new Vue({})）就能满足我们的使用场景了，而且也认为应该只存在一个。但是在课程里老师说（Vue2和Vue3启动不同）：有一个页面的多个应用的情况。所以想了解下，什么情况下会遇到这种场景呢？这个问题不是这节课的主题，但是还是很好奇，谢谢大圣老师解惑。

展开

2 9

Condor Hero 民

2021-10-25

讲的很细，很赞👍，有个小需求，希望大圣老师加上推荐阅读这样类似的章节。

作者回复: 阅读推荐指的是文章合集吗？

2 4

码农初上路
2021-10-25

大圣好，vue2.7最低也能支持到IE9吗？

展开

作者回复: 对的，Vue2基于Object.defineProperty来做，兼容不了IE8

2

淡若清风过
2021-10-26

拐了，还以为已经录制好的视频拿来卖，看来是到明年都更新不完

作者回复: 极客时间的更新策略，也算是帮你制定了一个学习计划吧

1

II
2021-10-25

全面拥抱Vue 3会有更好的待遇，敲敲黑板，重点😎

展开

作者回复: 那是必须的 工资驱动

1

球球
2021-10-25

vue2升级成vue3 对应用到的element ui 也要升级是吗

作者回复: 是这样的，可以选择element-plus或者element3

1 1

人生如戏

2021-10-26

vue2.7 Composition API、vue2原来语法可以同时混用吗？因为有个项目大部分模块我在做，但有一个模块分给其他同事了，我个人想用 Composition API

展开

作者回复: 肯定是可以的，Composition是一个额外的功能可以选用

起風了

2021-10-26

老师好,手头上有个项目,是基于vue2的ant design pro vue来迭代的,还使用了一些其他例如表格组件库等,都是vue2的. 如果要升级vue3的话,使用@vue/compat的同时,其他的那些插件或组件库也要升级vue3版本对应的?如果其他插件或组件库没有vue3版本的是不是升级就比较困难或者不可行了?

展开

作者回复: @vue/compat可以处理大部分场景，但是组件库是个例外，因为很多组件库对Vnode本身的格式都做了定制，很难自动化升级， antd-vue现在也全面支持vue3了 直接换最新版本也不难的

南山

2021-10-26

打卡
目前部门的老项目用的@compositon/api包使用vue3的composition api来重构和优化，新项目直接使用vue3来开发
开阔眼界了，使用ast语法编译来转换vue2代码，真是一劳永逸！

展开

作者回复: 看来是用了gogocode，是不是非常的哇塞

速冻
2021-10-26

虽然公司一直在使用vue3，但还是能学到很多，蟹蟹大圣老师

作者回复: 快分享一下Vue3的经验

Mserke
2021-10-25

思考题：有需要升级的地方，项目越来越大了，要拆分成微服务

@
2021-10-25

我们的项目也需要升级了

展开

作者回复: 升级完一定很有心得 欢迎回来分享

无双
2021-10-25

麻烦问一下，在哪有vue2的支持周期？

展开

作者回复: Vue3的全球发布会上，尤雨溪ppt里，Vue2还会支持18个月

Aybuai
2021-10-25

就我目前的项目来说有两点需要
1、Composition API 组合语法，项目过于庞大，有的页面上千行。维护起来特别麻烦。
2、自定义渲染器，随着项目的外推，可能不单单公司内部使用，可能会出现各种多端支持。

展开